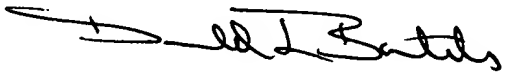
REMARKS

The foregoing Preliminary Amendment is submitted under 37 C.F.R. §1.115. This preliminary amendment amends the claims, specification and abstract to correct minor grammatical and typographical errors, and to provide clearer headings and further clarification to the claims. Applicants respectfully submit that no new matter is entered by any of the amended claims or by the other requested corrections, and entry of this preliminary amendment is respectfully requested.

Attached hereto is a marked-up version of the changes made to the abstract and claims by this preliminary amendment. The attached pages are captioned "**VERSION WITH MARKINGS TO SHOW CHANGES.**"

Respectfully submitted,



Donald L. Bartels
Registration No. 28,282

June 22, 2001
COUDERT BROTHERS
Four Embarcadero Center, Suite 3300
San Francisco, CA 94111
(415) 986-1300 (tel)
(415) 986-0320 (fax)

VERSION WITH MARKINGS TO SHOW CHANGES

IN THE SPECIFICATION:

The paragraph beginning at page 1, lines 4-5, has been amended as follows:

The present invention relates to packet switching, more specifically to data stream decoding and data stream ~~analysing~~ analyzing.

The paragraph beginning at page 1, lines 7-11, has been amended as follows:

~~Related art~~ Background of the invention

In the field of data and computer communications there is an increasing need for high speed/high bandwidth products.

~~Documents~~ Prior art references relating to packet switching and more specifically to data stream decoding and pertinent to the present invention includes:

The paragraph beginning at page 1, lines 32-39, has been amended as follows:

The invention relates to a device for data stream ~~analysing~~ analyzing. Said device is able to ~~recognise~~ recognize different data streams and then start other processors or functionalities to store or check data in a data stream. Special features are: a compare processor, a compare instruction memory, a data stream pipeline, a multiplexer and a multiplexer control unit, making it possible to test packet data under program control using several instructions and under several clock cycles even though ~~said~~ the data is moving

forward in the pipeline and even though other bytes of data are entering the device.

The paragraph beginning at page 2, line 6, has been amended as follows:

Fig. 2 is a block diagram of the multiplexer control unit of Fig. 1.

The underlined paragraph heading beginning at page 2, line 9, has been amended as follows:

Detailed description of ~~preferred embodiments~~ the invention

The paragraph beginning at page 2, lines 10-12, has been amended as follows:

The invention is preferably implemented as an integrated circuit (IC) having an electrical interface to the outside. The invention comprises a number of physical or logical units as illustrated in Fig. 1, including:

The paragraph beginning at page 2, line 35-page 3, line 2, has been amended as follows:

When the compare processor 5 ~~have~~ has come to such kind of conclusion it might want to report something to a result field or an option field, see below. This is done by starting up a save sequence. A start address for a save sequence will be sent from the compare processor 5 to the save engine 6. Said save engine 6 examines the incoming address and decides if it is a save regarding the result field or the option field. According to this decision the address is placed in either a bit save fifo register 61 or a stream save fifo register 62 respectively.

The paragraph beginning at page 3, lines 4-5, has been amended as follows:

The bit save unit 7 has three ~~functions;~~ functions: it can set bits in the result field, perform checksum control and length control.

The paragraph beginning at page 3, lines 10-26, has been amended as follows:

The delayline 1 preferably comprises a 23 shifts deep, 1 byte wide shift register. The 16 first positions of the shift register ~~are~~ is reachable from the compare processor 5 through a multiplexer 2. The two last positions are connected to the two save units 7,9 (bit save and stream save). The stream save unit 9 is actually only using the very last position, and only the bit save unit 7 needs the last two positions because the checksum control works with 16 bits at a time. There are five positions that are prevented from being accessed by the parsing function of the compare processor 5 and by the save units 7, 9 (bit save and stream save). The reason for a delay before the byte stream arrives to the save units 7, 9 is that all start addresses sent from the compare processor 5 to the save units 7, 9 are queued in a fifo register. Depending on how many save sequences in the queue and how long they are, this might in some extreme situations generate an error. This is because vital data already have passed through the delayline before a save sequence is started. The actual delay needed to secure that no such error occurs is 4*64=192 clock cycles. 64 is the maximum length of a save sequence and 4 is the maximum of start addresses waiting to

be executed. However, ~~calculus~~ simulations have ~~showed~~ shown that five delay cycles ~~is~~ are enough, since all save sequences normally written are very short.

The paragraph beginning at page 3, line 28-37, has been amended as follows:

A characteristic function of the invention is that it automatically keeps track of where a specific byte has its location in the delayline. The programmer only needs to specify which tag, i.e. which number the byte has, where the first byte in a packet is number zero, the second is number 1 and so on. This is why every byte arriving to the delayline 1 should be tagged (numbered). The tagging operating could easily be done by just adding an extra field in every shift in the delayline 1 inheriting the byte's tag. But his is disadvantageous in two aspects. First, much silicon would be used to implement the extra field in the delayline 1. Second, when the parser wants to look at a specific tag it would take a lot of time if every shift had to be searched to find the wanted tag.

The paragraph beginning at page 4, lines 1-10, has been amended as follows:

Instead, the present invention has solved the above problem by making a part of the delayline multiplexable; said multiplexable part of the delayline ~~comprises~~ preferably includes the 16 latest incoming bytes. Worst case for the length of a packet is 1 byte (erroneous), but since the first 12 bytes always contain the OSI Media Access Control address (MAC-address), no useful information can be extracted if the

packet is shorter than 13 bytes. These packets will force the compare processor 5 to begin with the next packet at once and their DV (data valid) signal will be unset so the rest of the device or a switch will never see it. With a limit of at least two clock cycles (bytes) between different packets it is possible to guarantee that never more than two packets exist at the same time in the delayline 1.

The paragraph beginning at page 4, lines 12-16, has been amended as follows:

According to the ethernet standard the IFG (Inter Frame Gap), which means the distance between packets, is at least 20 cycles, but a smaller distance is always desirable. E.g., a minumum distance of 6 cycles makes it possible to easily extend the device to be able to take care of SONET frames (An alternative ISO-028 Layer 2 frame instead of ethernet.

The paragraph beginning at page 4, lines 18-28, has been amended as follows:

~~The~~ Fig. 2 illustrates that the multiplex control unit 3 uses two identical Tag Units 32, 33 (TU), one for each possible packet, a Controlling Statemachine 31 (CS) to control the TU:s 32, 33 and a TU multiplexer 34 5o choose which one of the TU:s 32, 33 that the compare processor 5 is interested in. One TU includes a tagfield register 321, and a lastfield register 322, some adders and a simple statemachine 323. The other TU 33 is identical. When a packet arrives, the tagfield register 321 starts to increment for every byte. When the DV signal becomes false again the tagfield register 321

stops counting and the lastfield register 322 starts to increment. The TU 32 sends an ~~'end_of_packet'~~ "end of packet" signal when the ~~lastfield register 322 reaches the number of shifts in the delayline 1.~~ If the packet was shorter than 13 bytes a ~~'too_short'~~ "too short" signal will be generated.

The paragraph beginning at page 5, lines 11-15, has been amended as follows:

~~A~~ Referring again to Fig. 1, a feature of the device according to the present invention is that the compare processor 5 and the compare instruction memory 4 together act as a programmable parser. The description of the full instruction set of said parser is not ~~part of this document~~ set forth herein, but some instruction types are mentioned below. The parser uses four registers 51, 52, 54, 55 to fulfil ~~it's~~ its tasks.

The paragraph beginning at page 5, lines 28-35, has been amended as follows:

All instructions are executed in one clockcycle from the compare instruction memory 4 which is of the double ported memory type. This features decreases the total amount of clock cycles needed for the compare procesor 5 to parse a packet, thereby decreasing the needed size of the delayline. Some instructions are able to start save sequences. Said instructions have a field that tells what address in the save instruction memory 8 ~~that~~ shall start the execution. Save address 0x00 will not generate a start of a save sequence.

The paragraph beginning at page 6, lines 6-14, has been amended as follows:

The save engine 6 takes the address sent from the compare processor 5 and determines if it is the start address of a bit save sequence or a byte stream sequence. ~~After this~~ Thereafter the address together with the current value of the base register 54 is put in the specific fifo 61, 62. The value of the base register 54 is needed for all save instructions that ~~is~~ are using tag numbers. When the device according to the invention is programmed, a constant is written to the save engine 6 to tell where bit save sequences ~~ends~~ end in the save instruction memory 8. This feature exists because it is hart to tell how many instructions are needed to the different parts and it is more expensive to map two memories than one twice as big.

The paragraph beginning at page 6, lines 21-26, has been amended as follows:

The checksum unit 73 executes ~~the~~ a checksum control instruction which performs a 16-bit one complement addition. The unit needs to know what tag to start the execution from (Tag) and how many bytes the checksum should cover (Length). If there are checksum errors (i.e. the sum differs from 0xFFFF) the unit writes to the result field 76. Further, this block needs the value of the base register 54 as it was when the compare processor 5 sent the start address of the current save sequence.

The paragraph beginning at page 6, lines 29-36, has been amended as follows:

The bit unit 74 executes a bit save command which bitwise ~~'xor'~~ "xor"-ise one selected byte in the result field 76 with the data field. In other words, all bits which are set in the data field will invert the corresponding bit in the result field 76. It is only possible to invert one specific bit one time per packet, this is because e.g. an OSI Layer 3 error could be found in many ways, but if the bit which indicates a Layer 3 error is set an even number of times, this would look like a correct Layer 3 packet in the result field 76. The address field ~~tells to~~ indicates which ~~one~~ byte, of the ~~total~~ three possible bytes in the result field ~~76~~ 76, ~~to write~~ should be written to.

The paragraph beginning at page 7, lines 1-12, has been amended as follows:

The length error unit 75 is the most complex unit and investigates lengths in a packet and is used with one or more length control instructions. In a network there might occur packets that ~~has~~ have been cut off. This causes many ~~sorts~~ kinds of errors, e.g. if layer 4 is shorter than two bytes the result field 76 should indicate Layer 3 error but not Layer 2 error. The length error ~~unit~~ unit 75 consists of two identical checkboxes and one controller. A checkbox needs to know at which tag to start the measurement from, what kind of comparison it is supposed to perform (more, less, equal or not equal) and what length to match this comparison to. If a checkbox

detects a length error, a field which is part of the instruction ~~tells to~~ indicates which ~~one~~ bit, of four possible bits in the results field ~~76~~ 76, ~~to write~~ should be written to. As with the checksum unit 73, this unit 75 also needs the value from the base register 54 as it was when the compare processor 5 sent the start address of the current save sequence.

The paragraph beginning at page 7, lines 21-28, has been amended as follows:

The electrical interface of a preferred embodiment of the invention to the outside world is described in conjunction with fig. 3. It includes an input interface and an output interface. The input interface of the invention includes nine input terminals for a synchronous, eight bit wide, serial data stream, and a data valid (DV) signal, both used by the data that should be decoded. The input interface also includes a programming interface that comprises an 8-bit address bus, and 18-bit data bus, a chip select and a write enable signal for programming the two instructions. These 28 input terminals are used to program the invention after power on.

The paragraph beginning at page 7, line 37-page 8, line 7, has been amended as follows:

A typical application for the present invention is for packet switching in a computer network together with a packet switch by extracting information, especially addresses, from the packet headers, because it is possible to test data using several instructions and under several clock cycles even though said data is moving forward in the delayline

(1) and even though other bytes of data ~~is~~ are entering the device.

One of the features of the invention is that the decoding of the ~~protocol is programmable. This is a major advantage because new or~~ different types of protocols can be handled by just reprogramming the device. There will be no need for changing the hardware. This could save time and money for companies responsible for providing, maintaining and updating network switches.

IN THE CLAIMS:

1. (Once Amended) A device for data stream ~~analysing,~~ analyzing, comprising a processor means ~~including~~ and a program memory making it possible to parse a data stream in a way that is controlled by an interchangeable program.

2. (Once Amended) A device according to claim 1, ~~also including~~ further comprising a multiplexable data stream delayline for receiving said data stream, and multiplexing means for connecting different parts of the data stream to said processor means.

3. (Once Amended) A device according to claim 2, wherein the multiplexing means include a multiplexing control means for automatically keeping track of where specific data is located in the delayline, making it possible to write programs for controlling the device that can start executing at any time after the data have arrived to the device, and without the need for starting execution at a specific time relative to when ~~a~~ the data stream was entering the device.

4. (Once Amended) A device according to claim 2, wherein said delayline comprises a 23 shift deep, 1 byte wide shift register.

5. (Once Amended) A device according to claim ~~2, where~~ 3, wherein the multiplexing control means automatically keeps track of where specific data is located in the delayline by the use of ~~position registers (named tagfield and lastfield) that changes~~ a first and second position register that change according to certain rules when a packet is forwarded in the ~~delay line~~ delayline.

6. (Once Amended) A device according to claim ~~4,~~ 5, wherein ~~the value~~ values of the position registers are changed in the following way; when a packet arrives, the ~~tagfield~~ first register starts to increment for every byte; when the packet has come to its end, ~~i.e~~ where the packets DV (data valid) signal becomes false again, the ~~tagfield~~ first register stops counting and the ~~lastfield~~ second register starts to increment.

7. (Once Amended) A device according to claim 5, which automatically keeps track of where specific data is located in the delayline, by the use of said dedicated position registers together with the use of ~~the~~ a formula

$$pP = \text{tagfield} + \text{lastfield} - \text{wanted\_tag}$$

and ~~"p"~~ "P" is the position of ~~the~~ a wanted byte in the delayline; "tagfield" is the value of the ~~tagfield~~ first register; "lastfield" is the value of the ~~lastfield~~ second register and "wanted_tag" is the position of ~~the~~ a wanted byte relative to the beginning of the packet.

8. (Once Amended) A device according to claim 1, ~~including~~ further comprising a plurality of registers for making logical and/or arithmetic operations on data-stream data, before an actual comparison of the data with other data is executed.

9. (Once Amended) A device according to claim 1, ~~including~~ further compromising a stack memory means which enables the writing of one or more programs with subroutines for reducing ~~the~~ a need ~~of~~ for large program memories.

10. (Once Amended) A device according to claim 1, ~~which includes~~ further comprising a base address register for the ~~compare~~ processor means to make it possible to reuse

code to ~~recognise~~ recognize a given pattern even it starts at different positions in ~~a~~ the data stream.

11. (Once Amended) A device according to claim 1, wherein the program memory is of double ported type.

IN THE ABSTRACT:

A device for data stream ~~analysing~~ analyzing that ~~are~~ is able to ~~recognise~~ recognize different data streams and then start processors or functionalities to store or check data in a data stream~~,~~ ~~comprising~~. The device includes a processor means ~~including~~ and a program memory, making it possible to parse a data stream in a way that is controlled by an interchangeable program. There will be no need for changing the hardware. This could save time and money for companies responsible for providing, maintaining and updating network switches. The device also includes a multiplexable data stream delayline for receiving ~~said~~ the data streams, and multiplexing means for connecting different parts of the data stream to ~~said~~ the processor.